8/30/04

2183/#

41

In the United States Patent and Trademark Office

Serial number 09/340,172
Application filed June 25, 1999
Applicant Derek Wong
Application Title Methods for Increasing Instruction-Level Parallelism in Microprocessors and Digital Systems
Examiner/GAU Eric Coleman / 2183

Mailed August 26, 2004
at San Jose, California

From:
Derek Wong
1341 Echo Valley Dr.
San Jose, CA 95120

To:
Assistant Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RE: Amendment D to patent application 09/340,172

Thank you for reviewing patent application 09/340,172.

In response to the office action letter regarding patent application 09/340,172 mailed on March 26, 2004, the applicant encloses an amended set of claims and this letter.

The amended set of claims has the following changes:

Claims 77, 78, 79, 80, 84, and 85 have been amended.

Claim 83 has been cancelled.

The applicant respectfully submits the following comments in response to the office action:

1. Claim 78 has been amended to correct a typographical error: "previuos" changed to "previous."

2. In regard to claim 78, the patent by Grochowski et al (Pat #6,367,004) has a filing date of December 31, 1998. The present patent application is a follow-up to provisional patent application #60/090,782 filed on June 26, 1998. The present application claims the priority date of June 26, 1998 of that provisional patent application. The applicant respectfully submits that the cited reference by Grochowski has a filing date later than the present application's priority date and therefore that claim 78 should be allowable. (If this is not in agreement with the patent examiner's opinion, the applicant suggests that a clarifying conversation available by phone may be helpful. The applicant can be reached at 408-927-7940.)

3. In the interest of making the claim set ready for allowance, claim 79 has been amended to be a dependent claim upon claim 44, which has been previously allowed.

    a. Claim 79. (currently amended) A processor device <u>as in claim 44</u> comprising a data cache and a data hit/miss history table to record the history of whether previous executions of memory access instructions were hits or misses in said data cache, whereby run-time behavior about data hit/misses can be recorded in order to help optimize subsequent instruction scheduling.

4. Claim 77 has been amended to be dependent upon claim 44 rather than claim 76. Claim 44 has been previously allowed. This is mainly to make claim 77 to be dependent upon the same claim as the dependency of claim 79.

    a. Claim 77. (currently amended) A processor device as in claim ~~76~~ 44 comprising a value prediction history table to record the history of success or failure of previous executions of value predictions, whereby run-time behavior about value

predictions can be recorded in order to help optimize subsequent instruction scheduling.

5. In regard to claim 80, Kenner (Pat #5,903,749) presents a memory conflict resolution unit 345 which has a table 390. The table 390 stores information about each memory fetch that is pre-loaded so that an ambiguous store which is later executed can be compared with the entries in the table to see if any memory conflicts have occurred. One embodiment of the table, ALAT, is presented in col. 16, line 13 to col. 17, line 60. As each ambiguous memory store instruction is executed, it is compared against the table to see if any pre-loaded memory fetches conflict with this store instruction. If a conflict has occurred, then the processor needs to perform operations to correct previous operations based on incorrectly pre-loaded data. The processor needs to invalidate or correct for the incorrect memory pre-fetch and any instructions executed/partially executed that depend on that fetch (See flowcharts in Fig. 4 through 6; col. 17, lines 42-47; and col. 12, lines 3-46). An embodiment of a circuit that operates with the table is presented in col. 17, line 16 through col. 18, line 65 (the memory conflict resolution unit is 705 and table is 710 in Fig. 7).

   a. In Kenner, the method of correcting for a conflict includes reloading the memory fetch to get the correct data. This data is then saved as prefetch data for later.

   b. Kenner's table 390 is really a mechanism for checking for run-time conflicts rather than mainly intending to keep track of the history of whether conflicts have occurred previously over a number of executions of the same memory instructions.

c. Kenner's method does not store much of a history. In fact, the history info is one bit flag per entry in ALAT. Kenner's ALAT contains memory conflict entries with a single bit flag where the flag equals 1 if the entry is valid and no conflict has occurred. The flag equals 0 if either the entry is invalid or a conflict has occurred (col 16, lines 35-45). Furthermore, the flag is actually used just to determine whether or not to use the corresponding table entry during conflict checks rather than as predictive of future conflicts.

d. It is not clear how Kenner's method could be extended to take advantage of multiple bits of history or use the history as a predictive indicator, as the action of correction and data reload is taken whenever a conflict occurs and instruction scheduling otherwise appears to proceed assuming no conflict will occur (col. 15 and 16).

e. In contrast, the present patent application describes methods of changing the scheduling order of instructions based on history tables such as an ambiguous memory conflict history table (see pages 44-47 of the specification (the referenced page numbers are based on the substitute version submitted on December 21, 2003)). The history is used to predict the likelihood of future memory conflicts which is then used to affect instruction scheduling decisions.

f. In order to clarify the distinction between the presently claimed invention in claim 80 and the prior art, claim 80 is amended as follows:

g. Claim 80 (currently amended): A processor device comprising an ambiguous memory conflict history table to record the history of whether previous executions of promoted ambiguous read instructions cause memory conflicts or not, wherein

said processor device uses the history recorded in said ambiguous memory conflict history table as a predictive indicator of the likelihood of memory conflicts in subsequent executions in order to affect the scheduling order of said promoted ambiguous read instructions relative to other instructions in subsequent executions, whereby run-time behavior about ambiguous memory conflicts can be recorded in order to help optimize subsequent instruction scheduling.

h. The applicant respectfully submits that amended claim 80 is allowable over the cited art.

6. In the interest of making the claim set ready for allowance, claim 83 is cancelled without prejudice.

7. In regard to claim 84, Liu et al (Patent 5,978,900) presents a mechanism of generating a dependency vector between the decode and schedule stages of processing an UOP (Risc-like internally-generated instruction) during its execution. Figure 3 in Liu shows how a UOP 315 is sent to the data dependency stage unit 325 which then generates a UOP/Dependency vector. This is then sent to the schedule stage unit 335.

   a. Liu does not present any method of storing the dependency vector for use in repeated executions of the UOP. In Liu, the dependency vector is generated each time that the UOP is executed for use in scheduling that particular execution of the UOP.

   b. In contrast, the present patent application describes storing the dependency information in the transformed code as dependency vectors or pointers (see pages 53-55 of the specification. (The referenced page numbers are based on the substitute version submitted on December 21, 2003.)) The transformed code is

stored in the instruction stream cache 104 and is available for possible repeated execution.

c. In order to clarify the distinction between the presently claimed invention in claim 84 and the prior art, claim 84 is amended as follows:

d. Claim 84. (currently amended) A processor device comprising means of executing instructions from an instruction set architecture wherein the instruction set architecture can explicitly note dependencies between instructions by using dependency vectors, and wherein said processor device stores said instructions including said dependency vectors for possible repeated execution.

8. In order to clarify the distinction between the presently claimed invention in claim 85 and the prior art Liu, claim 85 is amended as follows:

a. Claim 85. (currently amended) A processor device comprising means of executing instructions from an instruction set architecture wherein the instruction set architecture can explicitly note dependencies between instructions by using dependency pointers, and wherein said processor device stores said instructions including said dependency pointers for possible repeated execution.

b. Additional note: Liu describes dependency vectors but does not appear to describe dependency pointers.

The applicant respectfully submits that the currently amended claims should be allowable over the prior art.

The applicant has reviewed the references cited in previous office actions. These references do not appear to describe the presently claimed inventions and do not appear render the claimed inventions to be obvious in any way.

The applicant is available at telephone number 408-927-7940 to discuss this application.

Thank you for your review of this application and thank you for your help.

Very respectfully,

Derek Wong

Patent Applicant

Express Mail Label # EU571761882US Date of Deposit August 26, 2004

I hereby certify that this paper or fee is being deposited with the United States Postal Service using "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to "Mail Stop Amendment, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450."

Signed _____ Inventor